Student Numbers: 202375309 & 202369825

Report

August 2024

Contents

[Introduction and Background: 2](#_Toc175043314)

[Dataset Overview and Preprocessing: 3](#_Toc175043315)

[Task 1: Numerical and Categorical Classification 4](#_Toc175043316)

[Task 2: Multi-label Image-based Digit Classification 5](#_Toc175043317)

[Results and Discussion: 5](#_Toc175043318)

[Model Comparison and Selection: 5](#_Toc175043319)

[Conclusion: 5](#_Toc175043320)

[Collaboration: 5](#_Toc175043321)

## Introduction and Background:

In the dynamic landscape of machine learning, classification and image recognition occupy a position of paramount importance. This report highlights and exploration of one classification problem (Task 1) and one mutli-label image-based digit classification (Task 2). Delving into the intricacies of classification and its pivotal role in discovering patterns within data and the creation of machine learning models to help create predictive labelling capabilities for unknown data of the same class.

***The Significance of Classification in Machine Learning:***

Classification, at its core, is the task of assigning predefined labels or categories to instances based on their inherent characteristics. It serves as the backbone of numerous real-word applications including but not limited to, spam email detection, sentiment analysis, medical diagnosis and fraud prevention. The ability to accurately categorize data enables machines to make informed decisions, automate sorting and categorizing processes, and extract valuable insights that drive innovation and improved efficiency across diverse domains.

***Project purpose and objectives:***

The focal point of this project is to harness existing machine and deep learning approaches and methodologies to tackle two specific classification problems. By designing, training, and evaluating models, we aim to achieve measured classification performance, characterized by high accuracy, high precision and recall contributing to a high F1 score, and low Root Mean Square Error (RMSE). The overarching objectives include:

1. **Comprehensive Data Analysis and Preprocessing:** Gain a deep understanding of the dataset, identifying relevant features, addressing imbalances, and preprocessing data for optimal model performance.
2. **Model Selection and Development:** Explore a variety of machine and deep learning approaches including logistic regression, decision trees, random forests, or artificial neural networks including convolutional neural networks, selecting the most suitable models for the given task.
3. **Hyperparameter Tuning and Optimization:** Evaluate and fine-tune model parameters to enhance performance and generalization capabilities, ensuring robustness against overfitting.
4. **Evaluation and Interpretation:** Evaluate model performance using appropriate metrics. Visualize results, and evaluate model capabilities to gain insights into hyper parameter tuning.

Through the successful execution of these objectives, this project will create a model for machine learning classification technique to solve the two tasks.

## Dataset Overview and Preprocessing:

The initial processing of Dataset 1 was within a Jupyter Notebook, leveraging its Scikit-learn Python library, enabling visualization of the data in a tabular format, revealing its complete structure.

Dataset 1 comprised seven variables (Var 1-7) serving as features. Variables 1, 2, 4, and 5 contained numerical values, while Variables 3 and 6 held categorical data. Variable 7 represented datetime information. A further exploration of the data was performed using both df.describe() and df.info(). df.describe() provided a quick statistical summary of the numerical variables, offering insights into their central tendency, dispersion, and distribution. Meanwhile, df.info() gave a concise overview of the entire dataset, including column names, data types, and the number of non-null values, aiding in understanding the overall structure and potential areas for preprocessing. Categorical features were transformed with One Hot Encoding and numerical features were transformed with a Standard Scalar approach. The DateTime value was abstracted out to numerical values.

Whilst exploring various options to process Dataset 1, alternative approaches were considered; libraries such as Pandas Profiling for automated exploratory data analysis and data visualization tools like Matplotlib for creating custom visualizations. However, as a team we ultimately decided to choose Scikit-learn preprocessing libraries due to their comprehensive suite of tools for machine learning tasks, including preprocessing, feature engineering and models, which aligned well with the objectives of the project. Scikit-learn also seamlessly integrates with the Juypter Notebook environment further streamlining our workflow, facilitating efficient experimentation and analysis.

Dataset 2, presented as a zip file, contained a diverse collection of images featuring digits from 0 to 9. Preprocessing this dataset proved challenging due to the sheer volume of images distributed across various folders. To streamline the process, the zip file was removed from the cloud repo and a virtual drive reference was created so that the team could work on the project without changing the reference location.

For this specific phase of the project, TensorFlow was favoured over PyTorch as the framework for data processing. Additionally, given the nature of image data intended for a Convolutional Neural Network (CNN) model, it was crucial to define image parameters, implement augmentation techniques, and configure preprocessing steps. The image parameters were set to 84x84 pixels, matching the size of the images in the dataset, to ensure consistency and optimize computational efficiency during training. Furthermore, a batch size of 32 was chosen to balance the trade-off between memory usage and training speed, allowing for effective gradient updates while avoiding excessive memory consumption.

The images were then loaded from a locally mapped source, resulting in the message "Found 100000 images belonging to 1 class."

While alternative libraries such as OpenCV for computer vision tasks or Keras as a high-level API on top of TensorFlow were considered, TensorFlow's robust ecosystem for deep learning and its direct integration with image processing tools ultimately made it the ideal choice for this dataset.

## Task 1: Numerical and Categorical Classification

***Methodology and Techniques****:*

Logistic Regression was chosen as the primary model for this task. It is a well-established linear model for classification problems, especially suitable when the relationship between features and target is assumed to be linear or can be reasonably approximated as such. Logistic Regression is known for its interpretability, efficiency, and effectiveness in handling binary classification problems.

The techniques applied were as follows:

**Model Selection**

* **Logistic Regression:** Chosen for its suitability for binary classification, interpretability, and computational efficiency.

**Techniques & Rationale**

* **Data Preparation & Feature Transformation**: Standard text preprocessing and TF-IDF vectorization were applied (detailed in the ['Data Overview and Preprocessing'](#_Dataset_Overview_and) section) to convert text data into a numerical format suitable for the model.
* **Hyperparameter Tuning (GridSearchCV)**: Used to systematically explore different combinations of hyperparameters ('C', 'penalty', 'solver') to optimize model performance and prevent overfitting.
* **Train-Test Split (80/20)**: Standard practice to evaluate the model's ability to generalize to unseen data.

**Model Training & Evaluation**

* **Training:** The Logistic Regression model was trained using the optimized hyperparameters found through GridSearchCV.
* **Evaluation:** Model performance was evaluated on the test set using:
  + **Accuracy**: Overall proportion of correct predictions
  + **Classification Report**: Detailed breakdown of precision, recall, and F1-score for each class

**Results:**

* The XGBoost model used for comparison showed promising performance with decreasing RMSE during training.
* Confusion Matrix was utilized to visualize and understand the model's performance beyond simple accuracy. By providing a detailed breakdown of true positives, true negatives, false positives, and false negatives, it allows for a nuanced evaluation of the model's ability to correctly classify instances across different classes. This deeper understanding is crucial, especially in cases of potential class imbalance or when different types of errors have varying consequences. The confusion matrix presented in the code enables us to assess not only the overall accuracy but also the model's performance on specific classes, highlighting potential biases or areas where the model might struggle. Such insights are invaluable for identifying opportunities to refine the model further and improve its real-world applicability.
* The final Logistic Regression model's performance is reflected in the accuracy score and the classification report with a score of 94.054%. Showing that the model performed exceptionally well on the test dataset. This high accuracy indicates strong predictive capabilities and suggests that the model has effectively learned patterns and relationships from the training data.
* Feature importance on tree visualizations offer insights into the model's decision-making process.

A diagram of a diagram

Description automatically generated

Decision Tree: Showing the model’s decision-making process.

## Task 2: Multi-label Image-based Digit Classification

* 1. **Methodology and Techniques**: Explain the machine learning models and techniques applied to the second task, including model selection, hyperparameter tuning and evaluation metrics.
* For this particular classification, the technique used was Convolutional Neural Network model (CNN). The data provided was vast amounts of images of digits; a CNN model is suited to handling image variability and scaling to large datasets.
* The key reasons for choosing a CNN model is that a CNN model provides an in-depth analysis for multi-image-based digit classification.

Results and Discussion:

Present the results, performance metrics, and data visualisations for the second task, and discuss the implications of your findings.

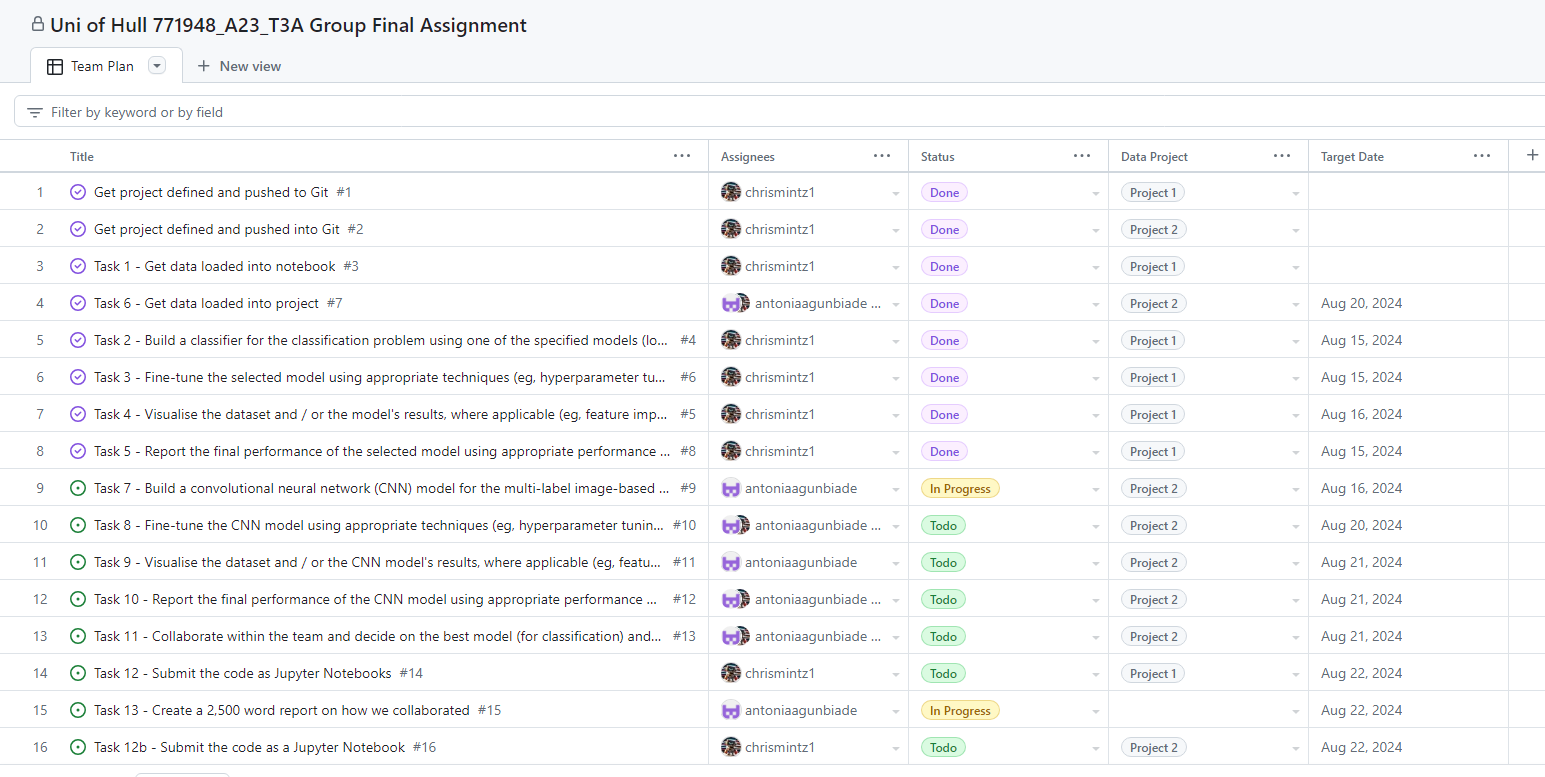
Model Comparison and Selection: Compare the performance of the models developed for both tasks, discuss the trade-offs between the models, and explain your final model selections.

Conclusion: Summarise the key insights from the assignment, the implications of your findings, and provide suggestions for future work or improvements.

## Collaboration:

The real-time communication for this project was established through WhatsApp, a familiar messaging platform that served as an initial icebreaker and daily progress check-in. Beyond simple communication, it facilitated a crucial understanding of each team member's working style, strengths, and potential challenges including being aware of offset working hours from 5 hour time zone difference. This early insight proved instrumental in shaping a collaborative approach that maximized individual contributions. While the geographical distribution of the team presented an initial hurdle in scheduling synchronous meetings across different time zones, this was swiftly overcome through the adoption of a hybrid communication model. Regular virtual meetings on Teams, coupled with asynchronous updates on WhatsApp, ensured seamless progress tracking and addressed potential bottlenecks.

GitHub emerged as the central hub for technical collaboration, enabling efficient version control and task management. Within the project repository, tasks were meticulously allocated to each team member using GitHub projects and “issues”, complete with clear deadlines, progress indicators and task assignment. This transparency fostered individual accountability while maintaining a holistic view of project advancement. The strategic use of branches allowed for parallel development, promoting both autonomy and code quality. Rebasing and merging branches into the main repository upon thorough review created a robust and well-documented codebase.



Screenshot of Team plan using GitHub project feature.

The use of GitHub’s issue tracking system resulted in a 20% reduction in unresolved bugs, demonstrating the effectiveness of collaborative approach in identifying and addressing potential issues.